Laurea Triennale in informatica - Università di Salerno
Corso di *Ingegneria del Software* - Prof. Carmine Gravino

TSR
Test Summary Report
Sapori di Unisa

Riferimento	nc01_saporidiunisa-tsr.docx
Versione	1.0
Data	01/02/2024
Destinatario	Prof. Carmine Gravino
Presentato da	Antonio Facchiano, Gianmarco Riviello, Salvatore Ruocco, Simone Vittoria
Approvato da	

Revision History

Data	Versione	Descrizione	Autori
27/01/2024	0.9	Stesura del documento	Gianmarco Riviello
01/02/2024	1.0	Revisione e delibera del documento	Facchiano Antonio; Riviello Gianmarco; Ruocco Salvatore; Vittoria Simone

Sommario

1 Introduzione 4
2 Relazione con altri documenti 4
3 Test di unità 5
4 Testing Coverage 5
5 Testing di sistema 6

1 Introduzione

Il documento di Test Plan ha l'obiettivo di descrivere e documentare il testing effettuato sulla piattaforma per poter garantire l'atteso funzionamento in ogni caso.

All'interno di questo documento sono riportate tutte le strategie scelte per effettuare il testing.

Sono stati effettuati i seguenti test per le varie Gestioni:

- Gestione Magazzino:
  - Registra fornitura
- Gestione Vendite:
  - Vendita prodotti
- Autenticazione:
  - Login
  - Modifica pin
- Gestione Scaffali:
  - Aggiungi prodotto scaffale
- Gestione Finanze:
  - Visualizza bilancio
  - Imposta sconto

2 Relazione con altri documenti

Questo documento va a completare la documentazione di tutto il testing effettuato prima del rilascio.

Gli altri documenti sono:

Test Plan

Nel Test Plan sono riportate tutte le scelte e le modalità riguardo la fase di testing.

Test Case Specification

Nel TCS sono documentati tutti i test case relativi alle funzionalità da testare.

Test Incident Report

Il Test Incident Report riporta gli esiti ottenuti durante l'esecuzione dei test progettati del Test Plan e specificati nel Test Case Specification.

3 Test di unità

Per i test di unità, si è scelto di verificare il funzionamento di 4 metodi dei controller, uno per programmatore, elencati di seguito e contenuti nel package dei test unit.

Nome	Tester	Esito
AddGiornoLavorativo	GR	Passed
EliminaLotto	SR	Passed
UpdatePin	AF	Passed
DiminuisciLotto	SV	Passed

4 Testing Coverage

Nel progetto i test sono stati suddivisi in vari package.

Ogni membro, prima del push, aveva l'obbligo di assicurarsi che tutti i test scritti riguardanti la classe appena modificata avessero successo. Dopo il push, tramite GitHub, vi è un sistema CI/CD integrato che esegue uno script che in caso di push con errore di sintassi manda una mail allo sviluppatore che ha effettuato il push avvertendolo del fatto che il codice ha errori.

In caso di problemi di questo tipo stava al programmatore che aveva appena effettuato il push correggere gli errori.

Nel corso della stesura del progetto si è scelto di utilizzare il tool *JaCoCo* per la raccolta delle metriche sul coverage dei test.

Branche Coverage	**Line Coverage**
48%	**50%**

Il report completo di JaCoCo è disponibile in docs/jacoco.

5 Testing di sistema

Ecco i risultati della configurazione Testing di Selenium.

Data di Esecuzione	#Fallimenti	#Successi
01/02/2024	0	29
01/02/2024	0	29

Importante: due dei test programmati, possono essere eseguiti solo modificando il tipo della data dal "developer" tool del browser, cosa che con Selenium IDE non è possibile; per questo sono stati eseguiti dagli sviluppatori, sia da browser che back-end con *Mockito* e aggiunti ai risultati.